PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 :
H04Q 3/00, H04L 12/24

A1

(11) International Publication Number: WO 00/47003

(43) International Publication Date: 10 August 2000 (10.08.00)

(21) International Application Number: PCT/US00/01623

(22) International Filing Date: 27 January 2000 (27.01.00)

(30) Priority Data:
09/236,111 25 January 1999 (25.01.99) US

(71) Applicant: MPATH INTERACTIVE, INC. [US/US]; 665 Clyde Avenue, Mountain View, CA 94043 (US).

(72) Inventors: ROTHSCHILD, Jeffrey, Jackiel; 15560 Old Ranch Road, Los Gatos, CA 95030 (US). RIEDEL, Andrew, D.; 3217 Upper Lock Avenue, Belmont, CA 94002 (US).

(74) Agents: RAY, Michael, B. et al.; Sterne, Kessler, Goldstein & Fox P.L.L.C., Suite 600, 1100 New York Avenue, Washington, DC 20005-3934 (US).

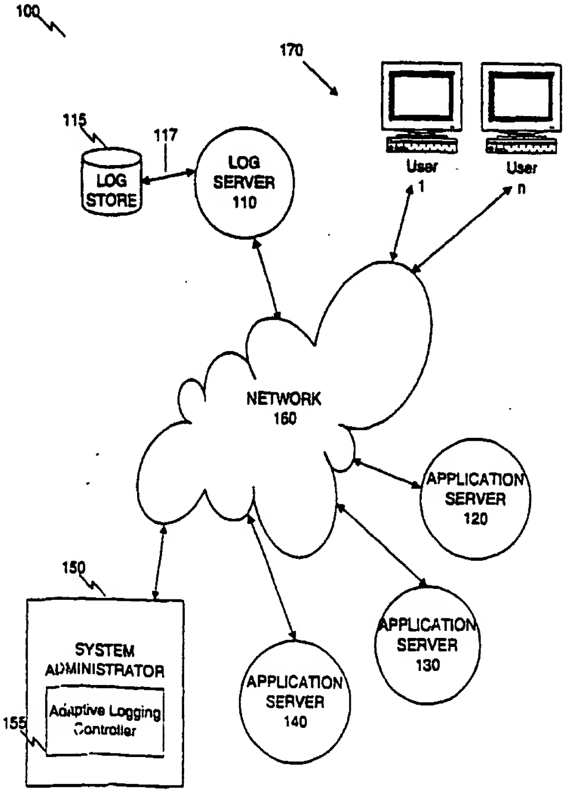
(81) Designated States: JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHOD, SYSTEM AND COMPUTER PROGRAM PRODUCT FOR ADAPTIVE LOGGING

(57) Abstract

A method, system, and computer program product provides adaptive logging. Logging events are filtered and stored locally based on at least one initial logging level. The stored logging events are flushed to a remote log server when a flush policy criteria is met. The initial logging level and flush policy criteria can be set or changed adaptively from a remote site. The initial logging level determines which priority levels of logging events are filtered and stored in a temporary local site. The flush policy determines when the filtered logging events are forwarded from the local site to a remote log server and persistent log store.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania
AM	Armenia
AT	Austria
AU	Australia
AZ	Azerbaijan
BA	Bosnia and Herzegovina
BB	Barbados
BE	Belgium
BF	Burkina Faso
BG	Bulgaria
BJ	Benin
BR	Brazil
BY	Belarus
CA	Canada
CF	Central African Republic
CG	Congo
CH	Switzerland
CI	Côte d'Ivoire
CM	Cameroon
CN	China
CU	Cuba
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
ES	Spain
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
GE	Georgia
GH	Ghana
GN	Guinea
GR	Greece
HU	Hungary
IE	Ireland
IL	Israel
IS	Iceland
IT	Italy
JP	Japan
KE	Kenya
KG	Kyrgyzstan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
KZ	Kazakstan
LC	Saint Lucia
LI	Liechtenstein
LK	Sri Lanka
LR	Liberia
LS	Lesotho
LT	Lithuania
LU	Luxembourg
LV	Latvia
MC	Monaco
MD	Republic of Moldova
MG	Madagascar
MK	The former Yugoslav Republic of Macedonia
ML	Mali
MN	Mongolia
MR	Mauritania
MW	Malawi
MX	Mexico
NE	Niger
NL	Netherlands
NO	Norway
NZ	New Zealand
PL	Poland
PT	Portugal
RO	Romania
RU	Russian Federation
SD	Sudan
SE	Sweden
SG	Singapore
SI	Slovenia
SK	Slovakia
SN	Senegal
SZ	Swaziland
TD	Chad
TG	Togo
TJ	Tajikistan
TM	Turkmenistan
TR	Turkey
TT	Trinidad and Tobago
UA	Ukraine
UG	Uganda
US	United States of America
UZ	Uzbekistan
VN	Viet Nam
YU	Yugoslavia
ZW	Zimbabwe

Method, System and Computer Program Product for Adaptive Logging

Background of the Invention

Field of the Invention

The present invention pertains to the field of network system administration. In particular, the present invention relates to obtaining, managing, and controlling log information on events in a network.

Related Art

Networks link many users. A distributed network, such as the Internet, links users from a variety of locations. Users can engage other users in real-time, interactive applications over a network. Examples of real-time, interactive applications include multi-player gaming applications and chat programs. On-line services further support real-time, interactive applications over a network. On-line services work to ensure a good user experience. For example, an on-line service, such as, MPLAYER offered by Mpath Interactive, Inc. at www.mplayer.com, supports matching, chat, and gaming facilities.

Logging information is generated during operation of an on-line service, and in particular, during the execution of an interactive application by multiple users. Logging information can be any type of information that is desired to be logged. For example, logging information can include any type of transactional and/or event log message generated during execution of an interactive gaming application. Logging information can include statistical, debug, and error information.

System administrators rely on logging information to monitor the performance of an on-line service and ensure a high quality user experience. System administrators can establish where logging will occur and what type of information will be logged. Such logging control, however, is typically limited to setting an initial logging level. Adaptive logging control responsive to real-time events is not provided.

For example, in an on-line service such as Mplayer, a system administrator module is connected to a log server and multiple application servers through a network. The application servers forward logging information to the log server for storage in a log store. Filters or loggers are provided at each application server to filter the logging information which is generated and sent to the log server. Through the system administrator module, a system administrator can send a command to the filters so that all application servers log all logging information including statistical, debug and error information. In this way, the administrator controls the level of logging. The system administration module also generates representations of the past or current performance of each application server. If a problem is reported at a particular application server, the system administrator can review stored logging information to diagnosis or trace the problem.

As the number of users increases and on-line service complexity increases, system administrators face an increasing amount of logging information. In setting an initial logging level, system administrators face a trade-off between the level of logging detail and the cost to store and review stored logging information that is collected. In a 10,000 user gaming service, if a system administrator sets a low logging level of detail to log all statistical, debug and error information then the amount of stored logging information can soon be quite large. Further this logging information is stored even when a user has a good experience. These factors increase both personnel costs and storage costs. Troubleshooting and error tracing becomes more time-consuming because a system administrator must sift through more irrelevant information. If the logging level is increased (i.e., increased to a higher level so that less information is logged) to log only error information, important information for debugging and statistical purposes is lost.

Efficient, adaptive logging is needed. A system administrator needs be able to collect sufficient logging information to identify when and where problems occur and what types of problems occur without having to collect an excessive amount of logging information based on an initial low logging level only.

Summary of the Invention

The present invention provides a method, system, and computer program product for adaptive logging. Logging events are filtered and stored locally based on at least one initial logging level. The stored logging events are flushed to a remote log server when a flush policy criteria is met. The initial logging level and flush policy criteria can be set or changed adaptively from a remote site. The initial logging level determines which priority levels of logging events are filtered and stored in a temporary local site. The flush policy determines when the filtered logging events are forwarded from the local site to a remote log server and persistent log store.

In one embodiment of the present invention, a system for adaptively logging at a node includes at least one class logger, a node logger, and at least one local log cache. Each class logger is coupled to at least one local log cache. The node logger is coupled between each local log cache and a remote log server. Each class logger filters logging events for a respective class of logging events based on at least one initial logging level and forwards the filtered logging events to a respective at least one local log cache for storage. The node logger determines whether a flush policy criteria is met. The node logger flushes the stored filtered logging events to the remote log server when the flush policy criteria is met.

An adaptive logging controller enables one to set or change at least one initial logging level for each respective class logger and to set or change a flush policy criteria for the node logger. In one example implementation, the adaptive logging controller sends commands to set or change the at least one logging level from a remote system administrator over the Internet to each class logger. In this way, the number of logging levels which are logged can be adaptively increased or decreased. According to a further feature, commands can be defined in a name space based on class names. The adaptive logging controller also sends commands to set or change the flush policy from the remote system administrator over the Internet to the node logger. In this way, the number of filtered logging events forwarded to a remote log server can be adaptively increased or decreased.

In one example implementation of the present invention, adaptive logging is provided in an on-line service linking multiple users over a network. The on-line service can support any interactive service or program, including but not limited to, chat and multi-player gaming applications. The network can be any public and/or proprietary network. In one embodiment, the on-line service includes an application server at a node in a network, such as, the Internet. The application server includes at least one class logger, a node logger, and at least one local log cache. The on-line service also includes a log server and a persistent log store. The log server is coupled over the network to the application server. A system administrator module includes an adaptive logging controller. The adaptive logging controller is coupled over the network to the application server. The adaptive logging controller enables one to set or change the initial logging level for each respective class logger and to set or change flush policy criteria for the node logger.

For example, the initial logging level can be set to log all debug, statistical, and error information locally at a particular node. The flush policy criteria can be set to flush the logged information only when 3 or more error messages or error logging events have been received in less than an hour. In this way, detailed logging information can be collected but if a user has a good gaming or chat experience (generating few or no error messages) the detailed information is discarded and does not hinder or waste the resources of the system log server or persistent log store. On the other hand, if the user experience is poor (generating more error messages), the flush policy criteria is met and the detailed logging information is forwarded to the log server and persistent log store. In this way, a system administrator can review the detailed log information in the persistent log store to troubleshoot and diagnosis problems. Further, the system administrator's burden is made easier as he or she is reviewing less irrelevant logging information from good gaming or chat experiences.

Further embodiments, features, and advantages of the present inventions, as well as the structure and operation of the various embodiments of the present invention, are described in detail below with reference to the accompanying drawings.

Brief Description of the Figures

The accompanying drawings, which are incorporated herein and form a part of the specification, illustrate the present invention and, together with the description, further serve to explain the principles of the invention and to enable a person skilled in the pertinent art to make and use the invention.

In the drawings:

FIG. 1 is a diagram of an example on-line service environment that supports adaptive logging according to the present invention.

FIG. 2 is a diagram of an application server with adaptive logging according to one embodiment of the present invention.

FIGs. 3A and 3B are flowcharts of a routine for adaptive logging according to one embodiment of the present invention.

FIG. 4 is an example log message.

FIG. 5 is a table of example levels of log messages.

FIG. 6 is an example computer system and computer program product in which the present invention is implemented primarily in software.

The present invention will now be described with reference to the accompanying drawings. In the drawings, like reference numbers indicate identical or functionally similar elements. Additionally, the left-most digit(s) of a reference number identifies the drawing in which the reference number first appears.

Detailed Description of the Preferred Embodiments

I. Overview and Discussion

The present invention provides a method, system, and computer program product for adaptive logging. The present invention is described in terms of an on-line service, such as, a multi-player gaming and chat service on the Internet. Description in these terms is provided for convenience only. It is not intended that the invention be limited to application in this example environment. In fact, after reading the following description, it will become apparent to a person skilled in the relevant art how to implement the invention in alternative environments known now or developed in the future.

II. Terminology

To more clearly delineate the present invention, an effort is made throughout the specification to adhere to the following term definitions as consistently as possible.

The terms "log event," "logging event," "log message," or "logging information" are used interchangeably to refer to any type of logging information or other information desired to be collected including, but not limited to, the example logging message shown in FIG. 4. Examples of logging information include debug, statistical, and error information.

The term "adaptive logging" refers to any logging performed according to the present invention that involves an initial logging level and/or a flush policy criteria. For example, logging is "adaptive" in that an initial logging level used to filter logging events can be set or changed in response to logging activity. Likewise, the flush policy criteria used to trigger the flushing of the filtered logging events can be set or changed in response to logging activity. Further, the flush policy criteria can be set or changed independent of the initial logging level.

III. Adaptive Logging

FIG. 1 shows an example network architecture 100 of an on-line service that supports adaptive logging according to the present invention. Architecture 100 includes log server 110, persistent log store 115, application servers 120, 130, 140, system administrator 150, and network 160. System administrator 150 includes an adaptive logging controller 155.

In accordance with one embodiment of the present invention, adaptive logging controller 155 allows a system administrator to set or change an initial logging level and to set or change flush policy criterion for forwarding local log information from an application server 120-140 to a persistent log store 115. A system administrator can set the initial logging level and flush policy for each application server, individually, in a distributed fashion. In this way, the system administrator can tailor or tune the amount and type of logging information collected locally and forwarded from each application server 120-140 to log server 110 for storage in log store 115. Moreover, the initial logging level can be changed independent of the flush policy criteria. Thus, the amount and type of information which is logged locally can be controlled independent of the criterion used to control the forwarding of the information to a persistent log store. This operation is described further with respect to FIGs. 2, 3A, and 3B.

Network 160 interconnects log server 110, log store 115, application servers 120, 130, 140, and system administrator 150. Log server 115 is also coupled to log server 110 through a local connection 117 (or through a link in network 160). One or more users 170 are coupled through network 160 to any of application servers 120, 130 or 140. Users 170 can be any type of end user device, including but not limited to, a personal computer, modem, telephone, television, set-top box, video game console, personal data assistant, or other terminal.

Each of the applications servers 120, 130, 140 and/or users 170 can execute one or more applications. For example, application server 130 can support one or more multi-player games. The functionality required to execute a game can be divided in any proportion between an application server and a user depending upon a particular gaming application and the available resources in application servers 120, 130, 140 and users 170. Users 170 can connect to application server 130 to play an interactive multi-player game over network 160. Network 160 can be any communication network (voice and/or data) or combination of networks including, but not limited to, the Internet. Users 170 can match themselves or be matched automatically by a separate matchmaker server (not shown). Users 170 can be coupled to one another through any communication link, including but not limited to, links over network 160. Links between users 170 can be client/server links through one or application servers 120, 130, 140. Peer-to-peer or direct links between users 170 can also be used.

Application servers 120-140 and users 170 each generate logging information. Logging information, also called logging messages, can be generated by the applications, servers, users, or any other network device. This logging information can include different types or levels of logging information. These levels can be prioritized depending upon impact upon system performance, user experience or other criteria. For example, levels of statistical, debug and error logging information can be prioritized to range from a low priority to a high priority.

FIG. 4 shows an example logging message 400. Logging message 400 includes four fields 402-408. Fields 402 and 404 are used to identify a log source. Global universal identification field 402 contains a global identifier number (GUID). Class field 404 identifies a particular class. Log event level field 406 identifies the priority level of the logging message, e.g., Debug 9, a ninth level of debugging. Log content field 408 contains logging information associated with the individual logging message 400. For example, log content field 408 can contain a text message "Service Problem" as shown in FIG. 4. FIG. 5 shows an example table of 20 priority levels of logging information including statistical, debug and error logging information. These example logging messages are illustrative and not intended to limit the present invention. In general, any type of logging message of fixed or variable length can be used in the present invention. A special name space can be used for defining and filtering logging information according to a separate feature of the present invention as described below.

FIG. 2 is a more detailed diagram of application server 130 with adaptive logging according to one embodiment of the present invention. Application server 130 includes client proxy 210, local log cache 215, property manager 230, class logger 240, node logger 250, and a name domain manager 260. Client proxy 210 is software or logic used to communicate with a user 170 to support the interactive applications hosted or supported by application server 130. Pieces of content 220 represent different functions in one or more active, interactive applications hosted or supported by application server 130.

An interactive application can include, but is not limited to, a multi-player matching service, game and/or chat program. For clarity, the present invention is described with respect to an application written in an object-oriented (OO) programming language such as Java. An OO language includes different functions or operations in classes. The present invention is not intended to be so limited, however, and can be applied to any programming language including procedural or other non-OO languages.

A class logger 240 is assigned for each class of the pieces of content 220. FIG. 2 only shows one class logger 240 for clarity. However, multiple class loggers 240 are used when multiple classes are involved. Property manager 230 in response to commands from system administrator 150 defines which pieces of content 220 are active and assigns class loggers 240. Class logger 240 receives and filters each logging event in a respective class. Class logger 240 then forwards the filtered logging events to local log cache 215. Node logger 250 reviews logging events and determines whether a flush policy criteria is met. Node logger 250 forwards the logging information stored in local log cache 215 to log server 110 when the flush policy criterion is met. Name domain manager 260 is used to control the registration of different names in a special name space based on class names according to a further feature of the present invention. Adaptive logging of the present invention, however, is not limited to use of a special name space and can be run without a name domain manager 260 or any particular name space. The operation of application server 130 and each of its components 210-260 is described further with respect to FIGs. 3A and 3B.

FIGs. 3A and 3B are flowcharts of a routine 300 for adaptive logging according to one embodiment of the present invention. Routine 300 includes steps 302- 380.

A. Initialization

Steps 302 and 304 relate to the initialization of adaptive logging according to the present invention. In step 302, initial logging levels are set for each class. Each class logger 240 is set to an initial respective logging level. The initial logging levels can be the same or different for different class loggers 240 depending upon which levels of logging information need to be logged for a particular class. In step 304, a flush policy is set. The node logger 250 is set to an initial flush policy criterion. Application server 130 can perform these steps in response to commands received from adaptive logging controller 155 or can initialize based on default values. For example, a system administrator, through the adaptive logging policy controller 155, can set an initial logging level of class logger 240 to "Debug 1" or the lowest priority level at which logging begins. Node logger 250 can be set to flush if 3 or more error messages are received in less than one hour.

B. Logging Activity

In step 310, logging activity occurs. The broken line between step 304 and 310 indicates this can be an asynchronous event depending upon when a piece of content 220 begins. As shown in FIG. 3B, when logging activity occurs, steps 320-380 are carried out in accordance with the initialization steps 302 and 304 performed earlier. In step 320, log events generated by a piece of content 220 are sent to class loggers 240. Class loggers 240 filter the log events based on the initial logging levels set in step 302 (step 330). Class loggers 240 forward all filtered log events which equal or exceed the initial logging level set in step 302 to a local log cache 215 (step 350). Alternatively, class loggers 240 can forward the filtered log events to node logger 250 for review and storage in local log cache 215.

In step 360, a check is made by node logger 250 to determine whether a flush policy criterion set in step 304 has been met. For example, an initial flush policy of three error messages within one hour might be set in step 304. Using this example, if node logger 250 determines that three or more error messages have been received in less than one hour in step 360, then node logger 250 flushes log cache 215 (step 380). In particular, the filtered log events stored locally in local log cache 215 are forwarded to log server 110. Log server 110 stores the filtered log event information in persistent log store 115.

If node logger 250 determines that a flush policy criterion has not been met in step360, node logger 250 checks for a change in flush policy (step 362). If no change in flush policy has been made, then the routine 300 discards the contents of local log cache 215 (step 364) and proceeds back to step 320 to process additional log events. If a change in flush policy has been made, then the flush policy of node logger 250 is updated (step 370). Routine 300 then proceeds to step 320, to continue processing log events according to the updated flush policy criterion. In general, step 364 can be performed after step 362 as described above, or after any indication of a positive user experience or expiration of a time interval that occurs when a flush policy criterion has not been met.

FIG. 6 is an example computer system and computer program product in which the present invention is implemented primarily in software.

Routine 300 and each of its constituent steps 302-380 can be implemented primarily in hardware, firmware, software, and any combination thereof. FIG. 2 is a block diagram of an embodiment of the present invention implemented primarily in an application server 130 according to the present invention. FIG. 6 is an example computer system 600 in which the present invention is implemented primarily in software.

An example of a computer system 600 is shown in FIG. 6. The computer system 600 represents any single or multi-processor computer. Single-threaded and multi-threaded computers can be used. Unified or distributed memory systems can be used.

The computer system 600 includes one or more processors, such as processor 604. One or more processors 604 can execute software implementing routine 300 as described above. Each processor 604 is connected to a communication infrastructure 602 (e.g., a communications bus, cross-bar, or network). Various software embodiments are described in terms of this exemplary computer system. After reading this description, it will become apparent to a person skilled in the relevant art how to implement the invention using other computer systems and/or computer architectures.

Computer system 600 also includes a main memory 608, preferably random access memory (RAM), and can also include a secondary memory 610. The secondary memory 610 can include, for example, a hard disk drive 612 and/or a removable storage drive 614, representing a floppy disk drive, a magnetic tape drive, an optical disk drive, etc. The removable storage drive 614 reads from and/or writes to a removable storage unit 618 in a well known manner. Removable storage unit 618 represents a floppy disk, magnetic tape, optical disk, etc., which is read by and written to by removable storage drive 614. As will be appreciated, the removable storage unit 618 includes a computer usable storage medium having stored therein computer software and/or data.

In alternative embodiments, secondary memory 610 may include other means for allowing computer programs or other instructions to be loaded into computer system 600. Such means can include, for example, a removable storage unit 622 and an interface 620. Examples can include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an EPROM, or PROM) and associated socket, and other removable storage units 622 and interfaces 620 which allow software and data to be transferred from the removable storage unit 622 to computer system 600.

Computer system 600 can also include a communications interface 624. Communications interface 624 allows software and data to be transferred between computer system 600 and external devices via communications path 626. Examples of communications interface 624 can include a modem, a network interface (such as Ethernet card), a communications port, etc. Software and data transferred via communications interface 624 are in the form of signals which can be electronic, electromagnetic, optical or other signals capable of being received by communications interface 624, via communications path 626. Note that communications interface 624 provides a means by which computer system 600 can interface to a network such as the Internet.

The present invention can be implemented using software running (that is, executing) in an environment similar to that described above with respect to FIG. 6. In this document, the term "computer program product" is used to generally refer to removable storage unit 618, a hard disk installed in hard disk drive 612, or a carrier wave carrying software over a communication path 626 (wireless link or cable) to communication interface 624. A computer useable medium can include magnetic media, optical media, or other recordable media, or media that transmits a carrier wave. These computer program products are means for providing software to computer system 600.

Computer programs (also called computer control logic) are stored in main memory 608 and/or secondary memory 610. Computer programs can also be received via communications interface 624. Such computer programs, when executed, enable the computer system 600 to perform the features of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 604 to perform the features of the present invention. Accordingly, such computer programs represent controllers of the computer system 600.

In an embodiment where the invention is implemented using software, the software may be stored in a computer program product and loaded into computer system 600 using removable storage drive 614, hard drive 612,or communications interface 624. Alternatively, the computer program product may be downloaded to computer system 600 over communications path 626. The control logic (software), when executed by the one or more processors 604, causes the processor(s) 604 to perform the functions of the invention as described herein.

In another embodiment, the invention is implemented primarily in firmware and/or hardware using, for example, hardware components such as application specific integrated circuits (ASICs). Implementation of a hardware state machine so as to perform the functions described herein will be apparent to persons skilled in the relevant art(s).

IV Special Name Space

A special name space can also be used to enable a system administrator to control the filtering of log events. In particular, a special name space can be used to formulate a command that identifies the initial log level. For example, a command can be sent from adaptive policy controller 155 that identifies an initial log level as described above with respect to FIG. 5.

Alternatively, according to a further feature, a command can identify a logging level using a refined name space based on the names of the classes (in other words, based on class names). Name domain manager 260 manages registration (addition and deletion) of names in the refined name space. Preferably, the name space includes many or all of the class names in use in the pieces of content 220. An asterisk or other symbol can be used to identify an open range of classes. For example, in the case of an on-line service, a command to set or change an initial logging level might be as follows:

com.companyname.servicename.servername.class1.*;

where "company name" can be any company name (i.e., Sony), "service name" can be any service name (i.e., games.jeopardy), "server name" can be any application server name (i.e., Mpath server 1), and "class1" can be any class (i.e., automatching class representing matchmaking content). The asterisk represents an open query that encompasses all other classes. In this example, the initial logging level will be all logging events for classes associated with the command:

"com.companyname.servicename.servername.class1.*".

This example is illustrative. As would be apparent to a person skilled in the art given this description other refined name spaces and examples could be used.

V. Conclusion

While specific embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. It will be understood by those skilled in the art that various changes in form and details may be made therein without departing from the spirit and scope of the invention as defined in the appended claims. Thus, the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What Is Claimed Is:

1. A method for adaptively logging, comprising the steps of:

(a) filtering logging events based on at least one initial logging level;

(b) storing the filtered logging events;

(c) determining whether the filtered logging events meet a flush policy criteria; and

(d) flushing the filtered logging events stored in step (b) when the flush policy criteria is met in step (c).

2. The method of claim 1, further comprising the steps of:

(i) setting the at least one initial logging level; and

(ii) setting the flush policy criteria.

3. The method of claim 2, wherein said step (i) comprises sending a first command to set the at least one logging level from a remote system over the Internet to a first server; and said step (ii) comprises sending a second command to set the flush policy from the remote system over the Internet to the first server.

4. The method of claim 3, further comprising the step of defining said first command in a name space based on class names.

5. The method of claim 1, further comprising the steps of:

(e) changing the flush policy criteria during logging activity; and

(f) flushing the filtered logging events stored in step (b) when the changed flush policy criteria is met; whereby, the number of filtered logging events forwarded to a remote log server can be adaptively increased or decreased.

6. The method of claim 1, wherein said flushing step (d) includes forwarding the filtered logging events stored in step (b) to a remote log store.

7. The method of claim 1, further comprising the steps of:

(e) changing the at least one initial logging level during logging activity; whereby, the number of logging levels passed through said filtering step (a) and stored in said step (b) can be adaptively increased or decreased.

8. The method of claim 1, wherein the logging events comprise classes of logging events, and wherein each of said steps (a) to (d) are performed at a first server remote from a log server, and said step (a) comprises filtering each class of logging events based on a respective initial logging level; said step (b) stores the filtered logging events on a per class basis; said step (c) stores the filtered logging events locally at the first server; and said step (d) comprises flushing the stored filtered logging events from the first server to the log server when the flush policy criteria is met for a respective class.

9. The method of claim 8, wherein the first server comprises an application server in an on-line, multi-player gaming or chat service, the application server executing a piece of content, and further comprising the step of generating the logging events during the execution of the piece of content prior to said filtering step (a).

10. A system for adaptively logging at a node, comprising:

at least one local log cache;

at least one class logger, coupled to said at least one local log cache, wherein each class logger filters logging events for a respective class of logging events based on at least one initial logging level and forwards the filtered logging events to a respective at least one local log cache for storage; and a node logger, coupled between each local log cache and a remote log server, wherein said node logger determines whether a flush policy criteria is met and flushes the stored filtered logging events to the remote log server when the flush policy criteria is met.

11. The system of claim 10, further comprising:

an adaptive logging controller that enables at least one initial logging level to be set for each respective class logger and to set a flush policy criteria for said node logger.

12. The system of claim 11, wherein the adaptive logging controller sends a first command to set the at least one logging level from a remote system administrator over the Internet to each class logger, and sends a second command to set the flush policy from the remote system administrator over the Internet to said node logger.

13. The system of claim 11, wherein said adaptive logging controller enables the flush policy criteria to be changed during logging activity such that said node logger flushes the stored filtered logging events when the changed flush policy criteria is met; whereby, the number of filtered logging events forwarded to a remote log server can be adaptively increased or decreased.

14. The system of claim 11, wherein said adaptive logging controller enables the at least one initial logging level at respective class loggers to be changed during logging activity such that each class logger filters said logging events based on the changed initial logging levels; whereby, the number of logging levels can be adaptively increased or decreased.

15. The system of claim 11, wherein the logging events comprise classes of logging events, and each class logger receives a respective class of logging events.

16. A system for adaptively logging, comprising:

(a) means for filtering logging events based on at least one initial logging level;

(b) means for storing the filtered logging events;

(c) means for determining whether the filtered logging events meet a flush policy criteria; and

(d) means for flushing the filtered logging events stored in said storing means (b) when the flush policy criteria is met as determined by said determining means (c).

17. The system of claim 16, further comprising:

(i) means for setting or changing the at least one initial logging level; and

(ii) means for setting or changing the flush policy criteria.

18. A computer program product comprising a computer useable medium having computer program logic for enabling at least one processor in a computer system to provide adaptive logging, said computer program logic comprising:

means for enabling the at least one processor to filter logging events based on at least one initial logging level;

means for enabling the at least one processor to forward the filtered logging events for storage;

means for enabling the at least one processor to determine whether the filtered logging events meet a flush policy criteria; and means for enabling the at least one processor to flush the filtered logging events when the flush policy criteria is met.

19. The computer program product of claim 18, further comprising:

(i) means for enabling the at least one processor to set or change the at least one initial logging level; and

(ii) means for enabling the at least one processor to set or change the flush policy criteria.

20. An on-line service linking multiple users over a network, comprising:

an application server at a node in the network;

an adaptive logging controller coupled over the network to said application server;

a log server coupled over the network to said application server; and

a log store coupled to said log server;

wherein said application server includes:

at least one local log cache;

at least one class logger, coupled to said at least one local log cache, wherein each class logger filters logging events for a respective class of logging events based on at least one initial logging level and forwards the filtered logging events to a respective at least one local log cache for storage; and

a node logger, coupled between each local log cache and said log server, wherein said node logger determines whether a flush policy criteria is met and flushes the stored filtered logging events to said log server when the flush policy criteria is met; and

wherein said adaptive logging controller enables said at least one initial logging level to be set or changed for each respective class logger and enables a flush policy criteria to be set or changed for said node logger.

21. A method for adaptive logging of logging events, comprising the steps of:

(a) setting or changing an initial logging level for the logging events; and

(b) setting or changing a flush policy for the logging events independent of said step (a).

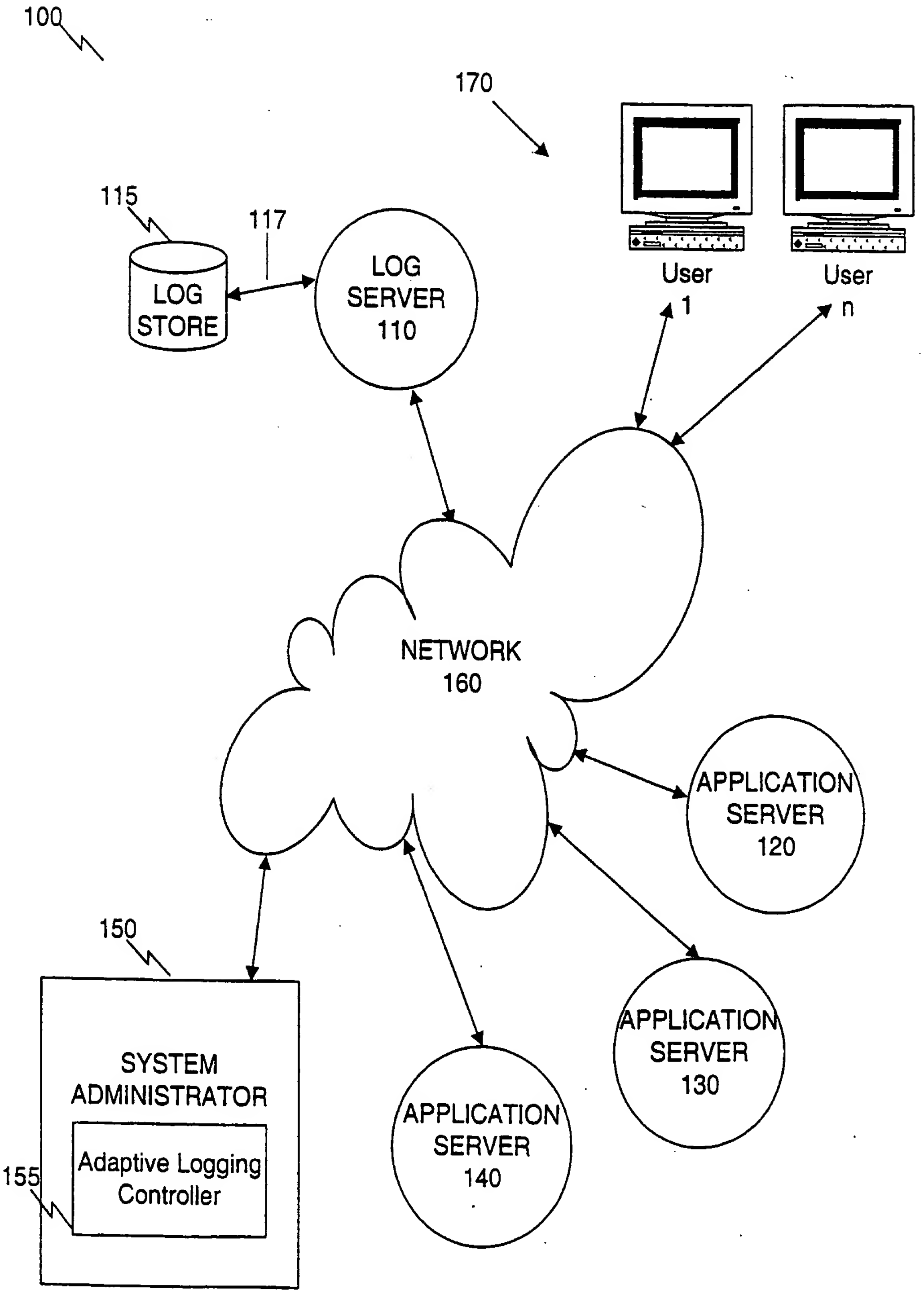

100
170
115
117
LOG STORE
LOG SERVER 110
User 1
User n
NETWORK 160
APPLICATION SERVER 120
APPLICATION SERVER 130
APPLICATION SERVER 140
150
SYSTEM ADMINISTRATOR
155
Adaptive Logging Controller

FIG. 1

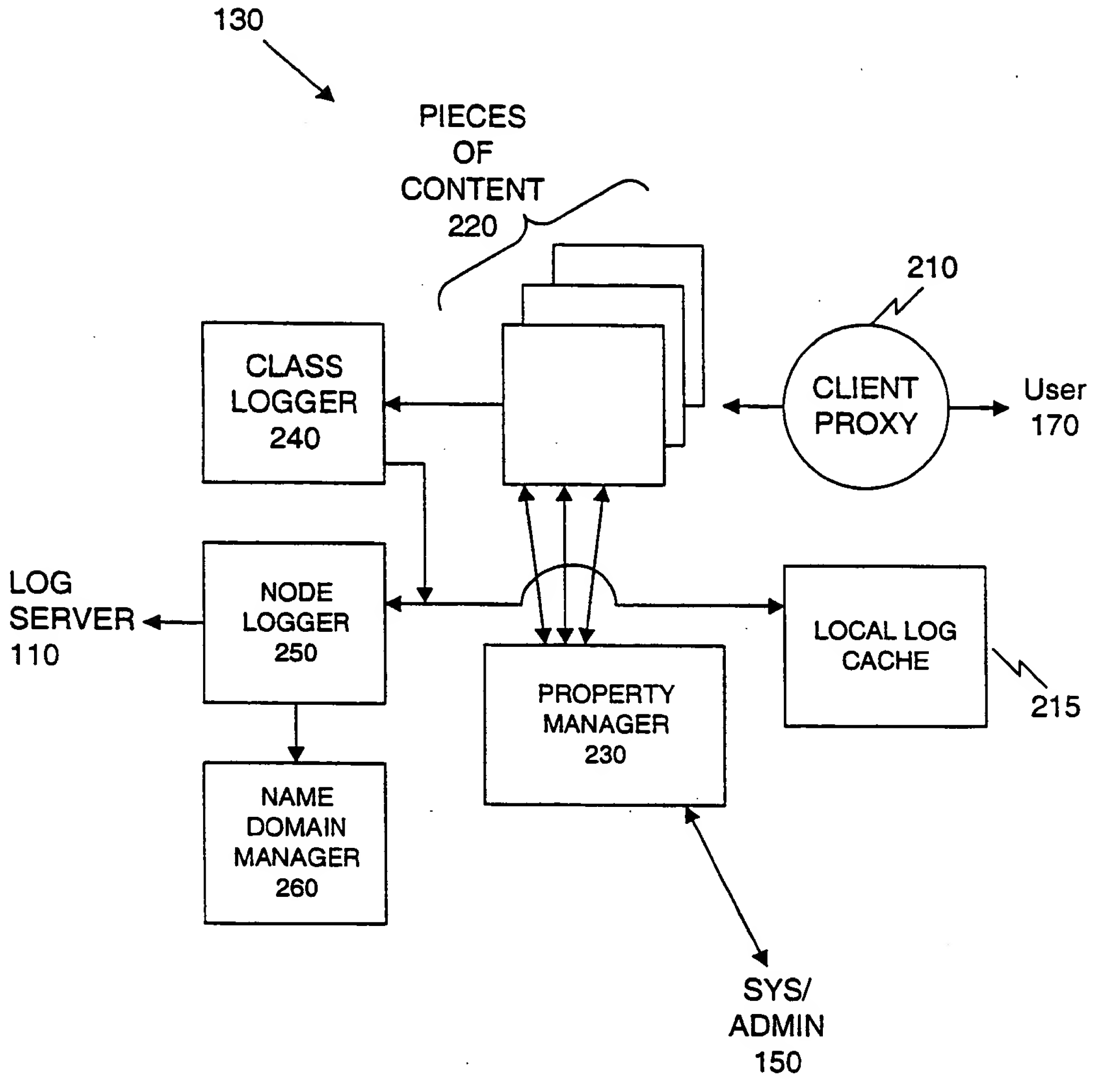

130
PIECES OF CONTENT 220
210
CLASS LOGGER 240
CLIENT PROXY
User 170
LOG SERVER 110
NODE LOGGER 250
LOCAL LOG CACHE
215
PROPERTY MANAGER 230
NAME DOMAIN MANAGER 260
SYS/ ADMIN 150

FIG. 2

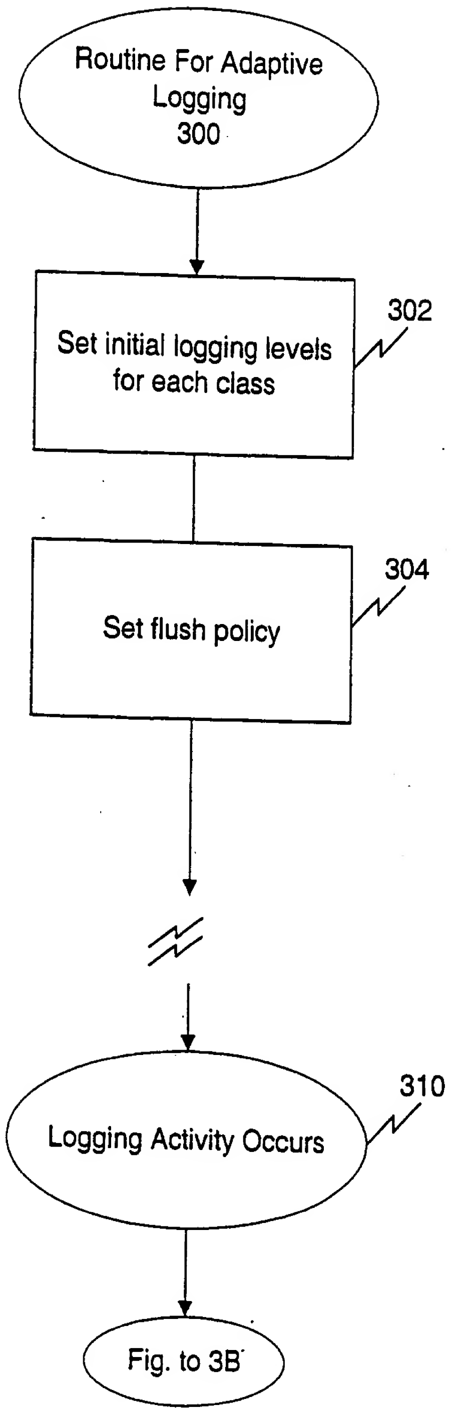

Routine For Adaptive Logging 300
Set initial logging levels for each class
302
Set flush policy
304
Logging Activity Occurs
310
Fig. to 3B

FIG. 3A

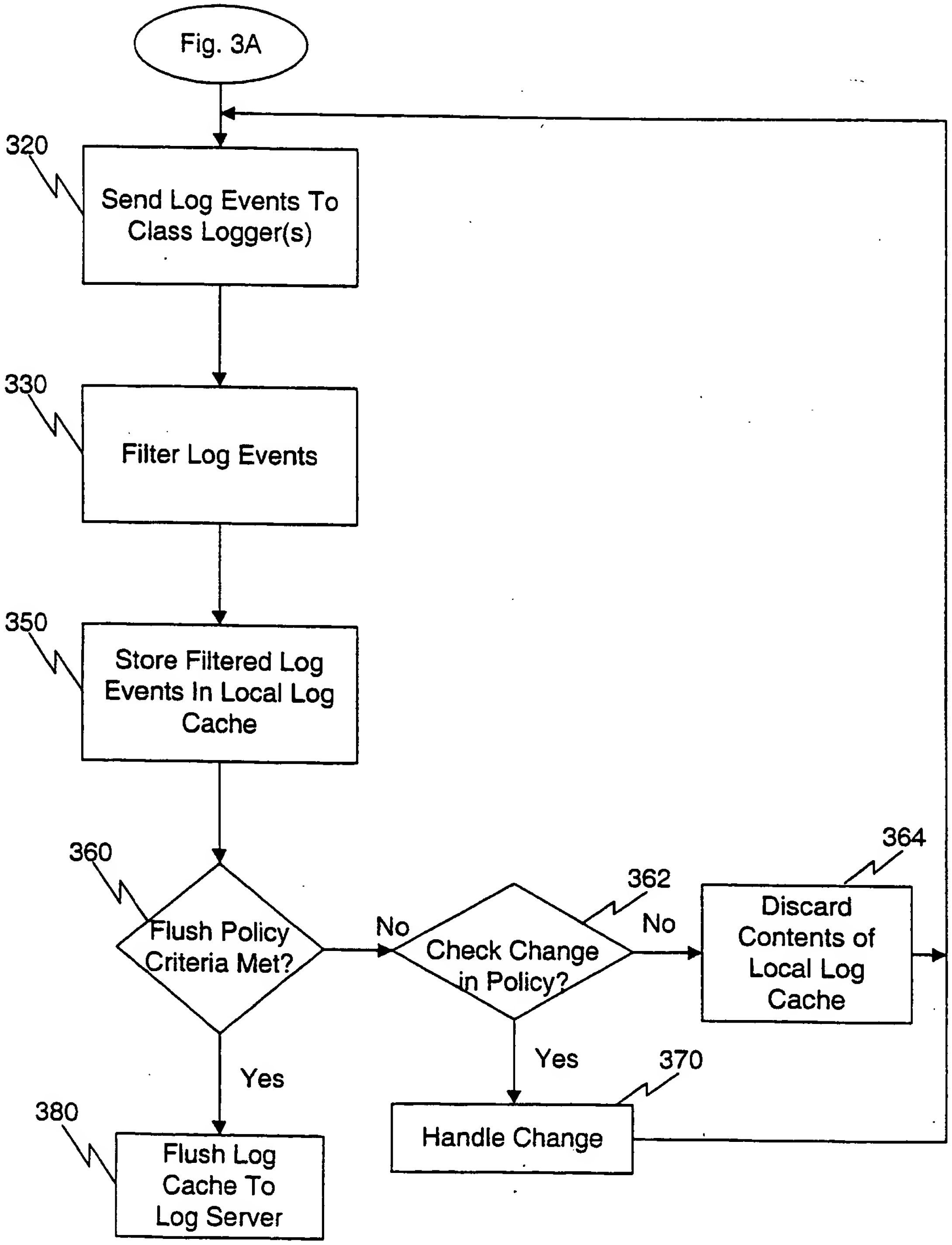


Fig. 3A
320
Send Log Events To Class Logger(s)
330
Filter Log Events
350
Store Filtered Log Events In Local Log Cache
360
Flush Policy Criteria Met?
No
362
Check Change in Policy?
No
364
Discard Contents of Local Log Cache
Yes
370
Handle Change
Yes
380
Flush Log Cache To Log Server

FIG. 3B

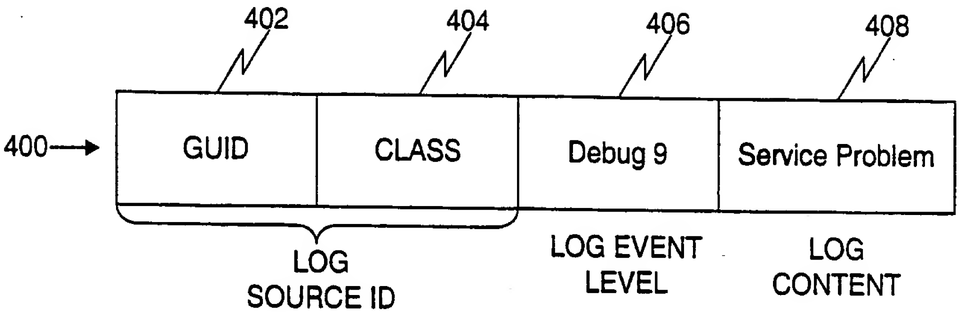
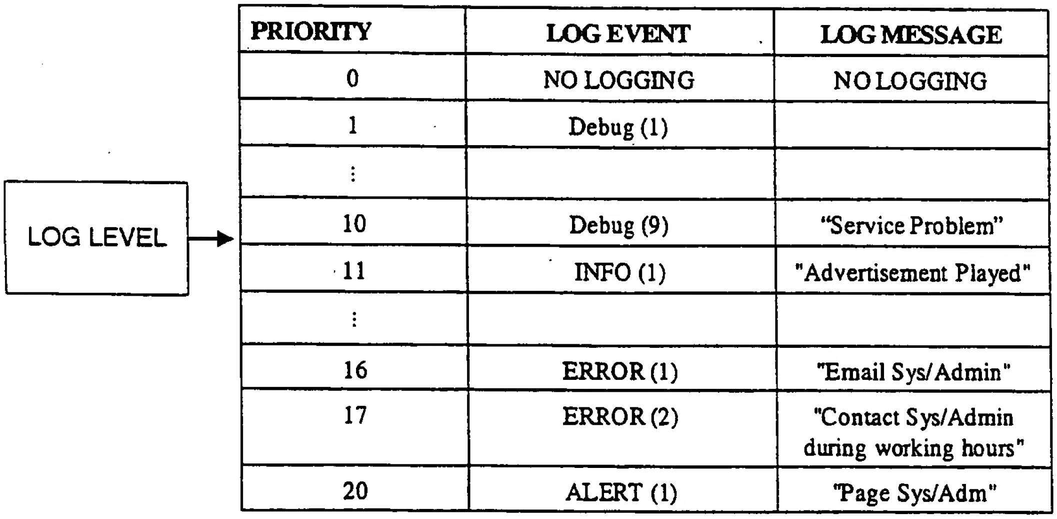

402
404
406
408
400
GUID
CLASS
Debug 9
Service Problem
LOG
SOURCE ID
LOG EVENT
LEVEL
LOG
CONTENT

FIG. 4



PRIORITY	LOG EVENT	LOG MESSAGE
0	NO LOGGING	NO LOGGING
1	Debug (1)	
⋮		
10	Debug (9)	"Service Problem"
11	INFO (1)	"Advertisement Played"
⋮		
16	ERROR (1)	"Email Sys/Admin"
17	ERROR (2)	"Contact Sys/Admin during working hours"
20	ALERT (1)	"Page Sys/Adm"

FIG. 5

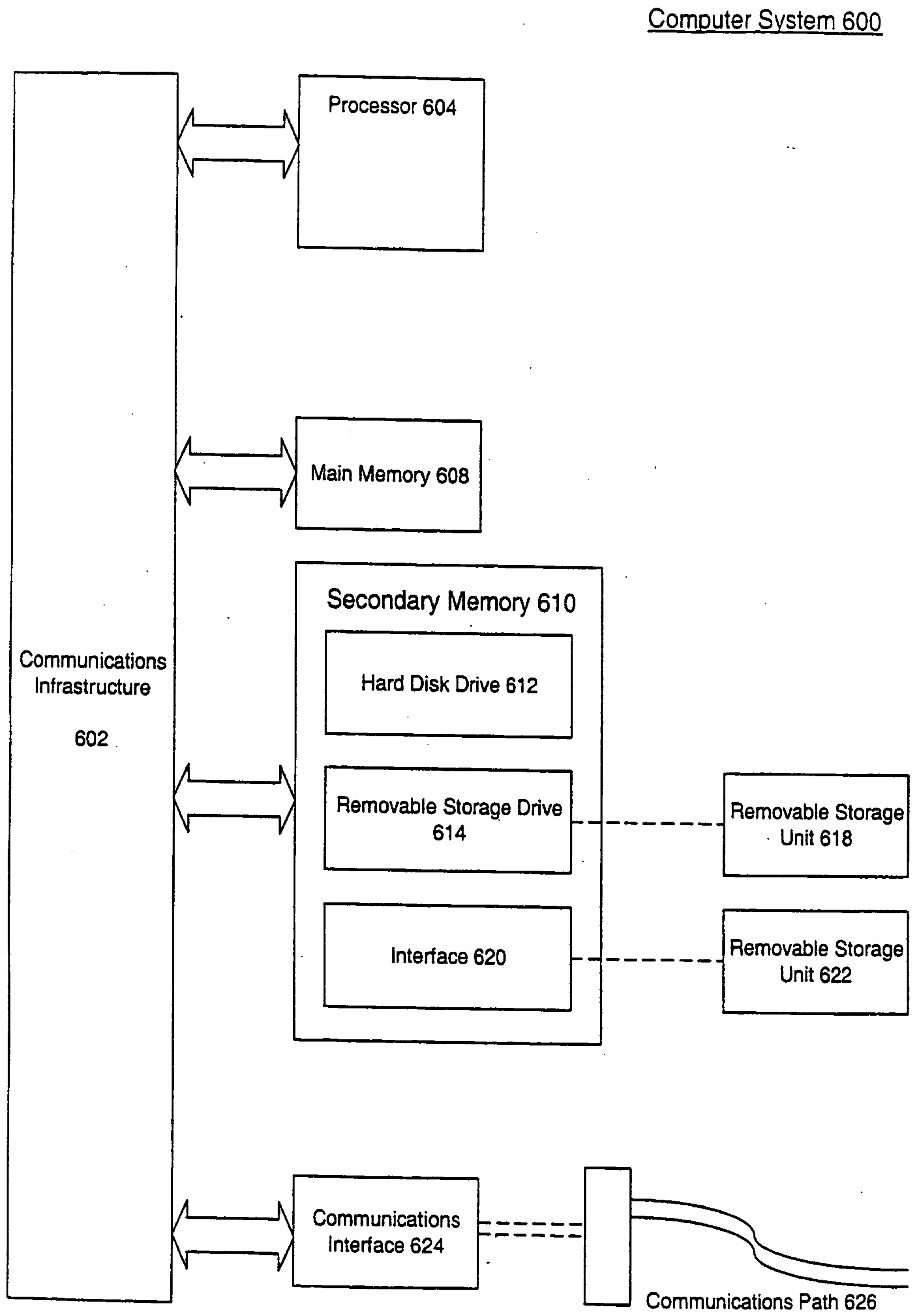

Computer System 600
Processor 604
Main Memory 608
Secondary Memory 610
Hard Disk Drive 612
Removable Storage Drive 614
Interface 620
Removable Storage Unit 618
Removable Storage Unit 622
Communications Infrastructure
602
Communications Interface 624
Communications Path 626

FIG. 6

INTERNATIONAL SEARCH REPORT

national Application No
PCT/US 00/01623

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04Q3/00 H04L12/24

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04Q H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5 857 190 A (BROWN M) 5 January 1999 (1999-01-05) column 2, line 22 - line 56 column 5, line 17 -column 7, line 25 column 7, line 54 - line 67 column 10, line 1 - line 25 column 12, line 10 -column 13, line 27; table 3	1,2, 5-11, 13-21
Y	— -/--	3,4,12

☒ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search
19 June 2000

Date of mailing of the international search report
28/06/2000

Name and mailing address of the ISA
European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer
Vercauteren, S

2

INTERNATIONAL SEARCH REPORT

national Application No
PCT/US 00/01623

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
Y	EP 0 759 591 A (IBM CORPORATION) 26 February 1997 (1997-02-26) page 5, line 11 - line 18 page 5, line 31 -page 6, column 9 page 7, line 2 - line 9 page 7, line 25 - line 29 page 11, line 53 -page 14	3,4,12
A		1,2, 5-11, 13-21
X	US 5 696 486 A (POLIQUIN L R ET AL) 9 December 1997 (1997-12-09) column 4, line 11 - line 24 column 7, line 20 - line 59 column 10, line 61 - line 67	1,2,6-8, 10,11, 14-21
A		3-5,9, 12,13
A	WO 96 24899 A (ERICSSON TELEFON AB LM) 15 August 1996 (1996-08-15) page 6, line 7 - line 15 page 8, line 3 - line 28 page 11, line 20 -page 12, line 37	1-21
A	US 5 655 081 A (BONNELL D N ET AL) 5 August 1997 (1997-08-05) column 9, line 21 -column 10, line 10 column 12, line 24 - line 31 column 13, line 6 -column 14, line 46	1-21

INTERNATIONAL SEARCH REPORT

Information on patent family members

national Application No
PCT/US 00/01623

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5857190 A	05-01-1999	NONE	
EP 0759591 A	26-02-1997	JP 9062523 A	07-03-1997
		JP 9062524 A	07-03-1997
US 5696486 A	09-12-1997	US 5777549 A	07-07-1998
		AU 5325896 A	16-10-1996
		EP 0818096 A	14-01-1998
		WO 9631035 A	03-10-1996
		US 6057757 A	02-05-2000
WO 9624899 A	15-08-1996	SE 504072 C	04-11-1996
		AU 692056 B	28-05-1998
		AU 4683596 A	27-08-1996
		BR 9607411 A	07-07-1998
		CA 2209912 A	15-08-1996
		CN 1173932 A	18-02-1998
		EP 0808488 A	26-11-1997
		FI 973913 A	08-10-1997
		JP 10513327 T	15-12-1998
		NZ 301420 A	28-07-1998
		SE 9500442 A	09-08-1996
US 5655081 A	05-08-1997	NONE	